UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/720,262	11/25/2003	Bjorn-Harald Sjogren	115719	4105

29078 7590 11/28/2006

CHRISTIAN D. ABEL
ONSAGERS AS
POSTBOKS 6963 ST. OLAVS PLASS
NORWAY, N-0130
NORWAY

EXAMINER	
DWIVEDI, MAHESH H	

ART UNIT	PAPER NUMBER
2168	

DATE MAILED: 11/28/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/720,262	SJOGREN ET AL.	
	Examiner	Art Unit	
	Mahesh H. Dwivedi	2168	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *26 October 2006*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *19-30* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *19-30* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *25 November 2003* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Response to Amendment

1. Receipt of Applicant's Amendment, filed on 10/26/2006, is acknowledged. The amendment includes amending the specification, the addition of a new abstract, the cancellation of Claims 1-18, and the addition of claims 19-30.

Specification

2. The objections raised in the office action mailed on 05/04/2006 have been overcome by the applicant's newly submitted abstract received on 10/26/2006.

Drawings

3. The objections raised in the office action mailed on 05/04/2006 have been overcome by the applicant's amendments received on 10/26/2006.

Claim Rejections - 35 USC § 112

4. The rejections raised in the office action mailed on 05/04/2006 have been overcome by the applicant's amendments received on 10/26/2006.

Claim Objections

5. Claims 19, 23, and 27 are objected to because of the following informalities: The examiner notes that the "bullet" points to several limitation should be removed. The examiner further suggests that applicant change "- to perform a delete process" to "A) to perform a delete process" as an example of an appropriate change. Appropriate correction is required.

Claims 20-22, 24-26, and 28-30 are objected to for incorporating the deficiencies of claim 19, 23, and 27.

Claim Rejections - 35 USC § 102

6. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

7. Claims 19-30 are rejected under 35 U.S.C. 102(b) as being anticipated by **Jacobs et al.** (U.S. Patent 6,105,025).

8. Regarding claim 19, **Jacobs** teaches a constraint enforcer comprising:

A) a set of constraints that governs the integrity of information stored in the data system (Column 7, lines 1-9, 43-64);

B) said enforcer being arranged to delay constraint checks until the end of a transaction by creating a check stack during the course of the transaction and executing entries on the check stack at the end of the transaction (Column 7, lines 1-9, 43-64, Column 9, lines 17-29, Column 10, lines 11-19);

C) said constraint enforcer comprising: a stack maker module, arranged for creating and updating said check stack (Column 9, lines 17-29, Column 10, lines 13-31);

D) said stack maker module being operatively connected to a runtime module in the database system (Column 6, lines 48-51, Column 9, lines 17-29); and

E) arranged to receive data from said runtime module (Column 6, lines 48-51, Column 9, lines 17-29);

F) said stack maker module being further arranged to perform a delete process on said check stack when said stack maker module is called upon as a result of a Delete or Update Data Manipulation Language operation (Column 10, lines 11-17, Figures 3A-3B);

G) to perform an insert process on said check stack when said stack maker module is called upon as a result of a Insert or Update Data Manipulation Language operation (Column 10, lines 11-17, Figures 3A-3B); and

H) to perform said delete process followed by said insert process when said stack maker module is called upon as a result of an Update Data Manipulation Language operation (Column 10, lines 11-17, Figures 3A-3B).

The examiner notes that **Jacobs** teaches **"a set of constraints that governs the integrity of information stored in the data system"** as "When a data value is created or added during a transaction or statement, a check is made to see if the new data vale has created a constraint violation. A constraint violation occurs when the new data value is inserted in a column having a uniqueness constraint and the value already exists in column" (Column 7, lines 1-6), "a list is generated for each uniqueness-required index for each session" (Column 9, lines 17-19), and "Uniqueness required index 308 is a B-tree structured index created on column 302N" (Column 10, lines 15-17). The examiner further notes that **Jacobs** teaches **"said enforcer being arranged to delay constraint checks until the end of a transaction by creating a check stack during the course of the transaction and executing entries on the check stack at the end of the transaction"** as enforcement may be deferred until processing is

completed for either a statement or transaction" (Column 7, lines 46-47), "constraint enforcement is deferred until the end of transaction (i.e., transaction level enforcement" (Column 7, lines 57-58). The examiner further notes that **Jacobs** teaches **"said constraint enforcer comprising: a stack maker module, arranged for creating and updating said check stack"** as "a list is generated for each uniqueness-required index for each session" (Column 9, lines 17-19), "insert, delete, and update operations" (Column 10, line 13), and "Uniqueness required index 308 is a B-tree structured index created on column 302N" (Column 10, lines 15-17). The examiner further notes that **Jacobs** teaches **"said stack maker module being operatively connected to a runtime module in the database system"** as "session (i.e., a connection between the application and the database system" (Column 6, lines 50-51) and "a list is generated for each uniqueness-required index for each session" (Column 9, lines 17-19). The examiner further notes that **Jacobs** teaches **"arranged to receive data from said runtime module"** as "session (i.e., a connection between the application and the database system" (Column 6, lines 50-51) and "a list is generated for each uniqueness-required index for each session" (Column 9, lines 17-19). The examiner further notes that **Jacobs** teaches **"said stack maker module being further arranged to perform a delete process on said check stack when said stack maker module is called upon as a result of a Delete or Update Data Manipulation Language operation"** as "Examples of insert, delete, and update operations using a uniqueness-required index are provided with reference to FIGS. 3A-3B" (Column 10, lines 13-15). The examiner further notes that **Jacobs** teaches **"to perform an insert process on said check**

stack when said stack maker module is called upon as a result of a Insert or Update Data Manipulation Language operation" as "Examples of insert, delete, and update operations using a uniqueness-required index are provided with reference to FIGS. 3A-3B" (Column 10, lines 13-15). The examiner further notes that **Jacobs** teaches **"to perform said delete process followed by said insert process when said stack maker module is called upon as a result of an Update Data Manipulation Language operation"** as "Examples of insert, delete, and update operations using a uniqueness-required index are provided with reference to FIGS. 3A-3B" (Column 10, lines 13-15).

Regarding claim 20, **Jacobs** further teaches a constraint enforcer comprising:

A) an enforcer module (Column 7, lines 1-9, 43-64);

B) arranged to receive check data from the check stack (Column 7, lines 1-9, 43-64);

C) to process the check data received from the check stack (Column 7, lines 1-9, 43-64); and

D) to provide resulting data to the runtime module (Column 6, lines 48-51, Column 9, lines 17-29).

The examiner notes that **Jacobs** teaches **"an enforcer module"** as "When a data value is created or added during a transaction or statement, a check is made to see if the new data vale has created a constraint violation" (Column 7, lines 1-3). The examiner further notes that **Jacobs** teaches **"arranged to receive check data from the check stack"** as "When a data value is created or added during a transaction or

statement, a check is made to see if the new data vale has created a constraint violation. A constraint violation occurs when the new data value is inserted in a column having a uniqueness constraint and the value already exists in column" (Column 7, lines 1-6). The examiner further notes that **Jacobs** teaches **"to process the check data received from the check stack"** as "When a data value is created or added during a transaction or statement, a check is made to see if the new data vale has created a constraint violation" (Column 7, lines 1-3) and "if constraint enforcement is deferred until the end of a transaction (i.e., transaction level enforcement)...are examined as an initial part of transaction commit processing" (Column 7, lines 57-62). The examiner further notes that **Jacobs** teaches **"to provide resulting data to the runtime module"** as "session (i.e., a connection between the application and the database system" (Column 6, lines 50-51) and "a list is generated for each uniqueness-required index for each session" (Column 9, lines 17-19). The examiner further notes that it common knowledge that during a session, information and data is exchanged between application and database programs.

Regarding claim 21, **Jacobs** further teaches a constraint enforcer comprising:

A) wherein said constraints are stored in a conceptual rules module (Column 6, lines 40-67);

B) comprising rules for prescribing permitted states and transitions that the database can undertake (Column 2, lines 65-67-Column 3, lines 1-16, Column 7, lines 1-6, Column 10, lines 15-17);

C) said conceptual rules module being operatively connected to said stack maker (Column 6, lines 48-51, Column 9, lines 17-29);

D) wherein said stack maker module is arranged to retrieve constraints from said conceptual rules module (Column 7, lines 1-6, Column 9, lines 17-29, and Column 10, lines 13-31).

The examiner notes that **Jacobs** teaches **"wherein said constraints are stored in a conceptual rules module"** as "the non-uniqueness count is maintained in dynamic memory. The storage for the non-uniqueness count is allocated when the first duplicate entry is added to the uniqueness-required index" (Column 6, lines 40-44) and "Constraint violations are tracked using the non-uniqueness count" (Column 6, lines 56-57). The examiner further notes that **Jacobs** teaches **"comprising rules for prescribing permitted states and transitions that the database can undertake"** as "Uniqueness-required index 308 is a B-tree structured index created on column 302N" (Column 10, lines 15-17), and "When a data value is created or added during a transaction or statement, a check is made to see if the new data vale has created a constraint violation. A constraint violation occurs when the new data value is inserted in a column having a uniqueness constraint and the value already exists in column" (Column 7, lines 1-6). The examiner further notes that **Jacobs** teaches **"said conceptual rules module being operatively connected to said stack maker"** as "session (i.e., a connection between the application and the database system" (Column 6, lines 50-51) and "a list is generated for each uniqueness-required index for each session" (Column 9, lines 17-19). The examiner further notes that it common

knowledge that during a session, information and data is exchanged between application and database programs. The examiner further notes that **Jacobs** teaches **"wherein said stack maker module is arranged to retrieve constraints from said conceptual rules module"** as "When a data value is created or added during a transaction or statement, a check is made to see if the new data vale has created a constraint violation. A constraint violation occurs when the new data value is inserted in a column having a uniqueness constraint and the value already exists in column" (Column 7, lines 1-6). The examiner further notes that it common knowledge that during a session, information and data is exchanged between application and database programs.

Regarding claim 22, **Jacobs** further teaches a constraint enforcer comprising: A) wherein said check stack is stored on persistent or volatile memory. (Column 10, lines 13-31, Column 16, lines 51-53, Figure 6).

The examiner notes that **Jacobs** teaches **"wherein said check stack is stored on persistent or volatile memory"** as "Uniqueness-required index 308 is a B-tree structured index created on column 302N" (Column 10, lines 15-17) and "The present invention can be implemented on a general purpose computer such as illustrated in FIG. 6...FIG. 6 also includes a video memory 614, main memory 615 and mass storage 612, all coupled to bi-directional system bus 618" (Column 16, lines 46-53).

Regarding claim 23, **Jacobs** teaches a method comprising:

A) a set of constraints that governs the integrity of information stored in the data system (Column 7, lines 1-9, 43-64);

B) comprising the step up delaying constraint checks until the end of a transaction by creating a check stack during the course of the transaction and executing entries on the check stack at the end of the transaction (Column 7, lines 1-9, 43-64, Column 9, lines 17-29, Column 10, lines 11-19);

C) performed by a stack maker module being operatively connected to a runtime module in said database system (Column 6, lines 48-51, Column 9, lines 17-29); and

D) receiving data from said runtime module (Column 6, lines 48-51, Column 9, lines 17-29);

E) creating and updating said check stack (Column 9, lines 17-29, Column 10, lines 13-31);

F) performing a delete process on said check stack when said stack maker module is called upon as a result of a Delete or Update Data Manipulation Language operation (Column 10, lines 11-17, Figures 3A-3B);

G) performing an insert process on said check stack when said stack maker module is called upon as a result of an Insert or Update Data Manipulation Language operation (Column 10, lines 11-17, Figures 3A-3B); and

H) performing said delete process followed by said insert process when said stack maker module is called upon as a result of a Update Data Manipulation Language operation (Column 10, lines 11-17, Figures 3A-3B).

The examiner notes that **Jacobs** teaches **"a set of constraints that governs the integrity of information stored in the data system"** as "When a data value is created or added during a transaction or statement, a check is made to see if the new data vale has created a constraint violation. A constraint violation occurs when the new data value is inserted in a column having a uniqueness constraint and the value already exists in column" (Column 7, lines 1-6), "a list is generated for each uniqueness-required index for each session" (Column 9, lines 17-19), and "Uniqueness required index 308 is a B-tree structured index created on column 302N" (Column 10, lines 15-17). The examiner further notes that **Jacobs** teaches **"comprising the step up delaying constraint checks until the end of a transaction by creating a check stack during the course of the transaction and executing entries on the check stack at the end of the transaction"** as enforcement may be deferred until processing is completed for either a statement or transaction" (Column 7, lines 46-47), "constraint enforcement is deferred until the end of transaction (i.e., transaction level enforcement" (Column 7, lines 57-58). The examiner further notes that **Jacobs** teaches **"performed by a stack maker module being operatively connected to a runtime module in said database system"** as "session (i.e., a connection between the application and the database system" (Column 6, lines 50-51) and "a list is generated for each uniqueness-required index for each session" (Column 9, lines 17-19). The examiner further notes that **Jacobs** teaches **"receiving data from said runtime module"** as "session (i.e., a connection between the application and the database system" (Column 6, lines 50-51) and "a list is generated for each uniqueness-required index for each session" (Column

9, lines 17-19). The examiner further notes that **Jacobs** teaches **"creating and updating said check stack"** as "a list is generated for each uniqueness-required index for each session" (Column 9, lines 17-19), "insert, delete, and update operations" (Column 10, line 13), and "Uniqueness required index 308 is a B-tree structured index created on column 302N" (Column 10, lines 15-17). The examiner further notes that **Jacobs** teaches **"performing a delete process on said check stack when said stack maker module is called upon as a result of a Delete or Update Data Manipulation Language operation"** as "Examples of insert, delete, and update operations using a uniqueness-required index are provided with reference to FIGS. 3A-3B" (Column 10, lines 13-15). The examiner further notes that **Jacobs** teaches **"performing an insert process on said check stack when said stack maker module is called upon as a result of an Insert or Update Data Manipulation Language operation"** as "Examples of insert, delete, and update operations using a uniqueness-required index are provided with reference to FIGS. 3A-3B" (Column 10, lines 13-15). The examiner further notes that **Jacobs** teaches **"performing said delete process followed by said insert process when said stack maker module is called upon as a result of a Update Data Manipulation Language operation"** as "Examples of insert, delete, and update operations using a uniqueness-required index are provided with reference to FIGS. 3A-3B" (Column 10, lines 13-15).

Regarding claim 24, **Jacobs** further teaches a method comprising:

A) performed by an enforce module (Column 7, lines 1-9, 43-64);

B) receiving check data from the check stack (Column 7, lines 1-9, 43-64);

C) processing the check data received from the check stack (Column 7, lines 1-9, 43-64); and

D) providing resulting data to the runtime module (Column 6, lines 48-51, Column 9, lines 17-29).

The examiner notes that **Jacobs** teaches **"performed by an enforcer module"** as "When a data value is created or added during a transaction or statement, a check is made to see if the new data vale has created a constraint violation" (Column 7, lines 1-3). The examiner further notes that **Jacobs** teaches **"receiving check data from the check stack"** as "When a data value is created or added during a transaction or statement, a check is made to see if the new data vale has created a constraint violation. A constraint violation occurs when the new data value is inserted in a column having a uniqueness constraint and the value already exists in column" (Column 7, lines 1-6). The examiner further notes that **Jacobs** teaches **"processing the check data received from the check stack"** as "When a data value is created or added during a transaction or statement, a check is made to see if the new data vale has created a constraint violation" (Column 7, lines 1-3) and "if constraint enforcement is deferred until the end of a transaction (i.e., transaction level enforcement)...are examined as an initial part of transaction commit processing" (Column 7, lines 57-62). The examiner further notes that **Jacobs** teaches **"providing resulting data to the runtime module"** as "session (i.e., a connection between the application and the database system" (Column 6, lines 50-51) and "a list is generated for each uniqueness-required index for each

session" (Column 9, lines 17-19). The examiner further notes that it common knowledge that during a session, information and data is exchanged between application and database programs.

Regarding claim 25, **Jacobs** further teaches a method comprising:

A) wherein said constraints are stored in a conceptual rules module (Column 6, lines 40-67);

B) comprising rules for prescribing permitted states and transitions that the database can undertake (Column 2, lines 65-67-Column 3, lines 1-16, Column 7, lines 1-6, Column 10, lines 15-17);

C) further comprising the step of retrieving by said stack maker module constraints from said conceptual rules module (Column 7, lines 1-6, Column 9, lines 17-29, and Column 10, lines 13-31).

The examiner notes that **Jacobs** teaches **"wherein said constraints are stored in a conceptual rules module"** as "the non-uniqueness count is maintained in dynamic memory. The storage for the non-uniqueness count is allocated when the first duplicate entry is added to the uniqueness-required index" (Column 6, lines 40-44) and "Constraint violations are tracked using the non-uniqueness count" (Column 6, lines 56-57). The examiner further notes that **Jacobs** teaches **"comprising rules for prescribing permitted states and transitions that the database can undertake"** as "Uniqueness-required index 308 is a B-tree structured index created on column 302N" (Column 10, lines 15-17), and "When a data value is created or added during a

transaction or statement, a check is made to see if the new data vale has created a constraint violation. A constraint violation occurs when the new data value is inserted in a column having a uniqueness constraint and the value already exists in column" (Column 7, lines 1-6). The examiner further notes that **Jacobs** teaches **"further comprising the step of retrieving by said stack maker module constraints from said conceptual rules module"** as "When a data value is created or added during a transaction or statement, a check is made to see if the new data vale has created a constraint violation. A constraint violation occurs when the new data value is inserted in a column having a uniqueness constraint and the value already exists in column" (Column 7, lines 1-6). The examiner further notes that it common knowledge that during a session, information and data is exchanged between application and database programs.

Regarding claim 26, **Jacobs** further teaches a method comprising:

A) wherein said check stack is stored on persistent or volatile memory. (Column 10, lines 13-31, Column 16, lines 51-53, Figure 6).

The examiner notes that **Jacobs** teaches **"wherein said check stack is stored on persistent or volatile memory"** as "Uniqueness-required index 308 is a B-tree structured index created on column 302N" (Column 10, lines 15-17) and "The present invention can be implemented on a general purpose computer such as illustrated in FIG. 6...FIG. 6 also includes a video memory 614, main memory 615 and mass storage 612, all coupled to bi-directional system bus 618" (Column 16, lines 46-53).

Regarding claim 27, **Jacobs** teaches a system comprising:

A) an application program interface, providing a two-way message interface to a user application program (Column 6, lines 48-51);

B) a runtime module, operatively connected to the application program interface (Column 6, lines 48-51, Column 9, lines 17-29);

C) a storage engine module, operatively connected to the runtime module (Column 16, lines 51-53);

D) a data storage, operatively connected to the storage engine module (Column 16, lines 51-53); and

E) a set of constraints that governs the integrity of information stored in the data system (Column 7, lines 1-9, 43-64);

F) said enforcer being arranged to delay constraint checks until the end of a transaction by creating a check stack during the course of the transaction and executing entries on the check stack at the end of the transaction (Column 7, lines 1-9, 43-64, Column 9, lines 17-29, Column 10, lines 11-19);

G) the constraint enforcer further comprising: a stack maker module, arranged for creating and updating said check stack (Column 9, lines 17-29, Column 10, lines 13-31);

H) said stack maker module being operatively connected to a runtime module in the database system (Column 6, lines 48-51, Column 9, lines 17-29); and

I) arranged to receive data from said runtime module (Column 6, lines 48-51, Column 9, lines 17-29);

J) performing a delete process on said check stack when said stack maker module is called upon as a result of a Delete or Update Data Manipulation Language operation (Column 10, lines 11-17, Figures 3A-3B);

K) performing an insert process on said check stack when said stack maker module is called upon as a result of an Insert or Update Data Manipulation Language operation (Column 10, lines 11-17, Figures 3A-3B); and

L) performing said delete process followed by said insert process when said stack maker module is called upon as a result of a Update Data Manipulation Language operation (Column 10, lines 11-17, Figures 3A-3B).

The examiner further notes that **Jacobs** teaches **"an application program interface, providing a two-way message interface to a user application program"** as "as "session (i.e., a connection between the application and the database system" (Column 6, lines 50-51). The examiner further notes that **Jacobs** teaches **"a runtime module, operatively connected to the application program interface"** as "session (i.e., a connection between the application and the database system" (Column 6, lines 50-51) and "a list is generated for each uniqueness-required index for each session" (Column 9, lines 17-19). The examiner further notes that **Jacobs** teaches **"a storage engine module, operatively connected to the runtime module"** as "a video memory 614, main memory 615 and mass storage 612 , all coupled to bi directional bus 618" (Column 16, lines 51-53). The examiner further notes that **Jacobs** teaches **"a data**

storage, operatively connected to the storage engine module" as "a video memory 614, main memory 615 and mass storage 612 , all coupled to bi directional bus 618" (Column 16, lines 51-53). The examiner further notes that **Jacobs** teaches **"a set of constraints that governs the integrity of information stored in the data system"** as "When a data value is created or added during a transaction or statement, a check is made to see if the new data vale has created a constraint violation. A constraint violation occurs when the new data value is inserted in a column having a uniqueness constraint and the value already exists in column" (Column 7, lines 1-6), "a list is generated for each uniqueness-required index for each session" (Column 9, lines 17-19), and "Uniqueness required index 308 is a B-tree structured index created on column 302N" (Column 10, lines 15-17). The examiner further notes that **Jacobs** teaches **"said enforcer being arranged to delay constraint checks until the end of a transaction by creating a check stack during the course of the transaction and executing entries on the check stack at the end of the transaction"** as enforcement may be deferred until processing is completed for either a statement or transaction" (Column 7, lines 46-47), "constraint enforcement is deferred until the end of transaction (i.e., transaction level enforcement" (Column 7, lines 57-58). The examiner further notes that **Jacobs** teaches **"the constraint enforcer further comprising: a stack maker module, arranged for creating and updating said check stack"** as "a list is generated for each uniqueness-required index for each session" (Column 9, lines 17-19), "insert, delete, and update operations" (Column 10, line 13), and "Uniqueness required index 308 is a B-tree structured index created on column 302N" (Column 10,

lines 15-17). The examiner further notes that **Jacobs** teaches **"said stack maker module being operatively connected to a runtime module in the database system"** as "session (i.e., a connection between the application and the database system" (Column 6, lines 50-51) and "a list is generated for each uniqueness-required index for each session" (Column 9, lines 17-19). The examiner further notes that **Jacobs** teaches **"arranged to receive data from said runtime module"** as "session (i.e., a connection between the application and the database system" (Column 6, lines 50-51) and "a list is generated for each uniqueness-required index for each session" (Column 9, lines 17-19). The examiner further notes that **Jacobs** teaches **"performing a delete process on said check stack when said stack maker module is called upon as a result of a Delete or Update Data Manipulation Language operation"** as "Examples of insert, delete, and update operations using a uniqueness-required index are provided with reference to FIGS. 3A-3B" (Column 10, lines 13-15). The examiner further notes that **Jacobs** teaches **"performing an insert process on said check stack when said stack maker module is called upon as a result of an Insert or Update Data Manipulation Language operation"** as "Examples of insert, delete, and update operations using a uniqueness-required index are provided with reference to FIGS. 3A-3B" (Column 10, lines 13-15). The examiner further notes that **Jacobs** teaches **"performing said delete process followed by said insert process when said stack maker module is called upon as a result of a Update Data Manipulation Language operation"** as "Examples of insert, delete, and update operations using a uniqueness-required index are provided with reference to FIGS. 3A-3B" (Column 10, lines 13-15).

Regarding claim 28, **Jacobs** further teaches a system comprising:

A) wherein said constraint enforcer further comprises an enforce module (Column 7, lines 1-9, 43-64);

B) arranged to receive check data from the check stack (Column 7, lines 1-9, 43-64);

C) to process the check data received from the check stack (Column 7, lines 1-9, 43-64); and

D) to provide resulting data to the runtime module (Column 6, lines 48-51, Column 9, lines 17-29).

The examiner notes that **Jacobs** teaches **"wherein said constraint enforcer further comprises an enforce module"** as "When a data value is created or added during a transaction or statement, a check is made to see if the new data vale has created a constraint violation" (Column 7, lines 1-3). The examiner further notes that **Jacobs** teaches **"arranged to receive check data from the check stack"** as "When a data value is created or added during a transaction or statement, a check is made to see if the new data vale has created a constraint violation. A constraint violation occurs when the new data value is inserted in a column having a uniqueness constraint and the value already exists in column" (Column 7, lines 1-6). The examiner further notes that **Jacobs** teaches **"to process the check data received from the check stack"** as "When a data value is created or added during a transaction or statement, a check is made to see if the new data vale has created a constraint violation" (Column 7, lines 1-3) and "if constraint enforcement is deferred until the end of a transaction (i.e.,

transaction level enforcement)...are examined as an initial part of transaction commit processing" (Column 7, lines 57-62). The examiner further notes that **Jacobs** teaches **"to provide resulting data to the runtime module"** as "session (i.e., a connection between the application and the database system" (Column 6, lines 50-51) and "a list is generated for each uniqueness-required index for each session" (Column 9, lines 17-19). The examiner further notes that it common knowledge that during a session, information and data is exchanged between application and database programs.

Regarding claim 29, **Jacobs** further teaches a system comprising:

A) wherein said constraints are stored in a conceptual rules module (Column 6, lines 40-67);

B). comprising rules for prescribing permitted states and transitions that the database can undertake (Column 2, lines 65-67-Column 3, lines 1-16, Column 7, lines 1-6, Column 10, lines 15-17);

C) said conceptual rules module being operatively connected to said stack maker (Column 6, lines 48-51, Column 9, lines 17-29);

D) said stack maker module being arranged to retrieve constraints from said conceptual rules module (Column 7, lines 1-6, Column 9, lines 17-29, and Column 10, lines 13-31).

The examiner notes that **Jacobs** teaches **"wherein said constraints are stored in a conceptual rules module"** as "the non-uniqueness count is maintained in dynamic memory. The storage for the non-uniqueness count is allocated when the first duplicate

entry is added to the uniqueness-required index" (Column 6, lines 40-44) and "Constraint violations are tracked using the non-uniqueness count" (Column 6, lines 56-57). The examiner further notes that **Jacobs** teaches **"comprising rules for prescribing permitted states and transitions that the database can undertake"** as "Uniqueness-required index 308 is a B-tree structured index created on column 302N" (Column 10, lines 15-17), and "When a data value is created or added during a transaction or statement, a check is made to see if the new data vale has created a constraint violation. A constraint violation occurs when the new data value is inserted in a column having a uniqueness constraint and the value already exists in column" (Column 7, lines 1-6). The examiner further notes that **Jacobs** teaches **"said conceptual rules module being operatively connected to said stack maker"** as "session (i.e., a connection between the application and the database system" (Column 6, lines 50-51) and "a list is generated for each uniqueness-required index for each session" (Column 9, lines 17-19). The examiner further notes that it common knowledge that during a session, information and data is exchanged between application and database programs. The examiner further notes that **Jacobs** teaches **"said stack maker module being arranged to retrieve constraints from said conceptual rules module"** as "When a data value is created or added during a transaction or statement, a check is made to see if the new data vale has created a constraint violation. A constraint violation occurs when the new data value is inserted in a column having a uniqueness constraint and the value already exists in column" (Column 7, lines 1-6). The examiner further notes that it common knowledge that

during a session, information and data is exchanged between application and database programs.

Regarding claim 30, **Jacobs** further teaches a system comprising:

A) wherein said check stack is stored on persistent or volatile memory. (Column 10, lines 13-31, Column 16, lines 51-53, Figure 6).

The examiner notes that **Jacobs** teaches **"wherein said check stack is stored on persistent or volatile memory"** as "Uniqueness-required index 308 is a B-tree structured index created on column 302N" (Column 10, lines 15-17) and "The present invention can be implemented on a general purpose computer such as illustrated in FIG. 6...FIG. 6 also includes a video memory 614, main memory 615 and mass storage 612, all coupled to bi-directional system bus 618" (Column 16, lines 46-53).

Response to Arguments

9. Applicant's arguments filed on 10/26/2006 have been fully considered but they are not persuasive.

Applicant goes on to argue on page 2, that **"It is respectfully submitted that the present invention relates to a transaction based constraint enforcer that is applicable with *any type* of constraints, such as primary keys, foreign keys, subset constraints, exclude constraints, etc. (cg. Par [0045])"**. However, the examiner wishes to point to Column 7 of **Jacobs** and refer to the first paragraph which states "When a data value is created or added during a transaction or statement, a check is made to see if the new data vale has created a constraint violation. A

constraint violation occurs when the new data value is inserted in a column having a uniqueness constraint and the value already exists in column" (Column 7, lines 1-6). The examiner further wishes to state that the claimed limitations in the instant application only claim "constraints". Clearly, a **Jacob's** method determines and uses constraints on a computer system. In response to applicant's argument that the references fail to show certain features of applicant's invention, it is noted that the features upon which applicant relies (i.e., "transaction constraints") are not recited in the rejected claim(s). Although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993).

Applicant goes on to argue on page 3, that **"In contrast, the entire disclosure of *Jacobs* appears to relate to the deferred enforcement of *uniqueness constraints* in particular"**. However, the examiner wishes to point to Column 7 of **Jacobs** and refer to the first paragraph which states "When a data value is created or added during a transaction or statement, a check is made to see if the new data vale has created a constraint violation. A constraint violation occurs when the new data value is inserted in a column having a uniqueness constraint and the value already exists in column" (Column 7, lines 1-6). The examiner further wishes to state that the claimed limitations in the instant application only claim "constraints". Clearly, a **Jacob's** method determines and uses constraints on a computer system. In response to applicant's argument that the references fail to show certain features of applicant's invention, it is noted that the features upon which applicant relies (i.e., "transaction

constraints") are not recited in the rejected claim(s). Although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993).

Applicant goes on to argue on page 3, that **"It is respectfully submitted that the above description does not imply that *Jacobs* discloses a stack maker module being operatively connected to a runtime module in the database system and arranged to receive data from the runtime module"**. However, the examiner wishes to point to Columns 9 and 10 of **Jacobs** which state "a list is generated for each uniqueness-required index for each session" (Column 9, lines 17-19), "insert, delete, and update operations" (Column 10, line 13), and "Uniqueness required index 308 is a B-tree structured index created on column 302N" (Column 10, lines 15-17). The examiner further wishes to state that it is common knowledge that a stack is an index structure used for storage/database operations.

Conclusion

10. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

U.S. Patent 5,706,494 issued to **Cochrane et al.** on 06 January 1998. The subject matter disclosed therein is pertinent to that of claims 1-18 (e.g., methods issue and enforce constraints on database systems).

U.S. Patent 6,453,314 issued to **Chan et al.** on 17 September 2002. The subject matter disclosed therein is pertinent to that of claims 1-18 (e.g., methods issue and enforce constraints on database systems).

U.S. Patent 5,408,657 issued to **Bigelow et al.** on 18 April 1995. The subject matter disclosed therein is pertinent to that of claims 1-18 (e.g., methods issue and enforce constraints on database systems).

11. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Contact Information

12. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Mahesh Dwivedi whose telephone number is (571) 272-2731. The examiner can normally be reached on Monday to Friday 8:20 am – 4:40 pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tim Vo can be reached (571) 272-3642. The fax number for the organization where this application or proceeding is assigned is (571) 273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Mahesh Dwivedi

Patent Examiner

Art Unit 2168

November 14, 2006

TIM VO
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

Leslie Wong

Primary Examiner